III. AMENDMENTS TO THE SPECIFICATION

Please amend the Title as follows:

~~PRE-DEPLOYMENT COMPONENT HOSTING ENVIRONMENT ANALYZER~~
INSTALLATION AND REMOVAL OF SOFTWARE COMPONENTS ACROSS ENTERPRISE RESOURCES

Please amend paragraph [1] as follows:

[1] This application incorporates by reference the following commonly-assigned and co-pending U.S. Patent Applications, filed on November 10, 2003: ~~IBM Docket Number RSW9-~~ United States Published Application 20050102665. 10/704,836~~2003-0175US1~~, entitled AUTOMATIC PARALLEL NON-DEPENDENT COMPONENT DEPLOYMENT; and ~~IBM Docket Number RSW9-2003-0177US1~~United States Published Application 20050102667, 10/705,555 entitled GENERATING SUMMARIES FOR SOFTWARE COMPONENT INSTALLATION.

Please amend paragraph [3] as follows:

[3] Many computer systems, a decade ago, employed hardware on which an operating system was installed to enable software applications to be run on the hardware. Fig. 1A illustrates such a simple configuration of hardware and software. More recently, however, businesses, governments, universities and others are taking advantage of large scale networks, including intranets and the internet, to allow users located virtually anywhere to easily access applications running on machines which are also located virtually anywhere. Thus, as illustrated in Fig. 1B, additional layers are required for a user at a browser client to ultimately (but transparently) access data through server-based applications. More importantly, such enterprise computing permits combining different, often incompatible, operating systems, applications and user interfaces into the same network.

Please amend paragraph [~~54~~] as follows:

[~~54~~] Large applications, such as application servers, may include hundreds or more individual components to install, each of which may include numerous sub-components. One example is the IBM® WebSphere® Application Server ("WAS"). In addition to the directories and files which comprise WAS, as illustrated in Fig. 2 WAS 200 also operates in conjunction with an object-oriented data base, such as IBM's DB2®-UDB 202, and an HTTP server, such as IBM's HTTP Server 204. Each of these applications ~~comprises~~ comprise many components and sub-components 206. Moreover, enterprise software is frequently deployed or installed in a cluster or group of machines. Thus, when the WAS Enterprise edition is deployed, components 206 of each of the three major components (WAS 200, DB2 202 and HTTP Server 204) are installed on many machines in order to achieve a satisfactory load balancing. Heretofore, such a deployment has been a labor intensive, time consuming and error prone activity by a system administrator installing many components across many machines in a domain. And, unfortunately, heretofore, such a deployment involves installing the files sequentially, thereby adding to the time required.

Please amend paragraph [5] as follows:

[5] An additional issue is raised due to the almost infinite number of combinations of software settings and configurations on multiple hosts with multiple parameters. Such complexity makes it extremely difficult for an administrator ~~is~~ to devise reliable test plans to insure the validity of change to software within an enterprise. Thus, seemingly harmless upgrades, patches, or new software may wreak havoc on an enterprise infrastructure. Existing software may unintentionally be compromised or corrupted by additional software or software updates. It will be appreciated that such unforeseen consequences may cause part or even all of a business's enterprise system to fail. For example, a new ~~Java~~JAVA Software Development Kit (SDK) is deployed each time an application, which uses ~~Java~~JAVA, is deployed. Although the ~~Java~~JAVA SDKs are supposed to be back-compatible they are not. Furthermore, developers commonly use both ~~Sun~~SUN and IBM ~~Java~~JAVA SDKs, introducing a number of incompatibilities. That is, ~~Java~~JAVA applications which were functional under ~~SUN~~SUN ~~Java~~JAVA version 1.3.1, for example, might not work properly under ~~SUN~~SUN ~~Java~~JAVA 1.4.1 or IBM ~~Java~~JAVA 1.3.1.

Please amend paragraph [6] as follows:

[6] The ~~Java~~JAVA SDK incompatibilities described above present one of the more common problems in ~~Java~~JAVA 2 Platform Enterprise Edition (J2EE) environments. However, although very harmful, this is a relatively simple problem to detect. More complicated problems are presented at the operating system (OS) and compiler levels. Frequently at the OS level there may be incompatibilities between different versions of an OS kernel and certain applications. For instance, IBM ~~Java~~JAVA SDK version 1.4.1 runs only with a ~~Linux~~LINUX kernel 2.2.5 or less, while the current ~~Linux~~LINUX kernel on ~~Redhat~~RED HAT ~~Linux~~LINUX is 2.5. Thus a new deployment will likely update the kernel and consequently perturb the functionality of the ~~Java~~JAVA Virtual Machine (JVM) and consequently all applications that use the JVM. A similar problem might occur with OS patches.

Please amend paragraph [7] as follows:

[7]More subtle problems may exist at the compiler level. Although different compilers use different optimization techniques, many developers are unaware of these techniques and the differences. Thus, a syntactically correct code may run differently on two compilers. For example, IBM employs the Just in Time Compilation technique (JIT) which provides an advanced optimization for the ~~Java~~JAVA code. Assume that certain code reads the time, then performs some computation and finally reads the time again. When an IBM compiler is used, the time difference between the two readings will be zero, because the compiler sees no dependency between the computation and the first time reading and thus will first execute the computation. In contrast, the same piece of code will run as intended using a ~~Sun~~SUN interpreter.

Please amend paragraph [26] as follows:

[26] The semantic model is a data structure stored in a knowledge base (as more fully described in commonly-assigned and co-pending U.S. Patent Serial Number 10/726,192, filed December 2, 2003, ~~IBM Disclosure Number RSW8-2003-0413,~~ entitled HOSTING ENVIRONMENT ABSTRACTION AGENTS, hereby incorporated by reference). The data structure need not be any particular structure; examples of possible structures include (but are not limited to) a flat file, a database, an object model, etc. The component semantic model is generated by the developer and may be bundled with the deployment package or accessed from a remote site during installation. In the event that deployment is to occur across domains, the model may be augmented with a list of target machines on which components will be installed.

Please amend paragraph [28] as follows:

[28] The present invention also identifies potential component conflicts by implementing a pre-deployment hosting environment analyzer. Again the semantic model for software components is employed which captures the topology of software components at different levels of detail as well as capturing complex relationships among components. The deployed components on the target are recorded in the eRegistry. The installation is as follows: as soon as an eReadme file is available to deploy (an eReadme captures the information about the components that are to be deployed), the eRegistry is examined and the knowledge base (as more fully described in commonly-assigned and co-pending U.S. Patent Serial Number 10/725,612, filed December 2, 2003, entitled OPTIMAL COMPONENT INSTALLATION) is accessed to download metadata about the relationship among the components to be installed and the components existing in the target. Next, the relationship data is analyzed so appropriate action may be taken in the event that a conflict is identified. For example, the installation may continue or the user may be alerted of the possible conflict. In the event installation continues, an entry may be recorded in a log for later reference. As soon the software is deployed on the target, the target eRegistry is updated with appropriate installation information.